Statistical Foundations of Prior-Data Fitted Networks

Thomas Nagler [1 2]

Abstract

Prior-data fitted networks (PFNs) were recently proposed as a new paradigm for machine learning. Instead of training the network to an observed training set, a fixed model is pre-trained offline on small, simulated training sets from a variety of tasks. The pre-trained model is then used to infer class probabilities in-context on fresh training sets with arbitrary size and distribution. Empirically, PFNs achieve state-of-the-art performance on tasks with similar size to the ones used in pre-training. Surprisingly, their accuracy further improves when passed larger data sets during inference. This article establishes a theoretical foundation for PFNs and illuminates the statistical mechanisms governing their behavior. While PFNs are motivated by Bayesian ideas, a purely frequentistic interpretation of PFNs as pre-tuned, but untrained predictors explains their behavior. A predictor's variance vanishes if its sensitivity to individual training samples does and the bias vanishes only if it is appropriately localized around the test feature. The transformer architecture used in current PFN implementations ensures only the former. These findings shall prove useful for designing architectures with favorable empirical behavior.

1. Introduction

1.1. PFNs in a Nutshell

Prior-data fitted networks (PFNs) were proposed by Müller et al. (2022) as a new approach to machine learning, motivated by ideas from Bayesian nonparametrics and meta-learning. The goal is to compute a posterior predictive distribution (PPD) for a test feature given observed training data. To approximate the PPD, a transformer network is trained offline through meta-learning. After simulating several training data sets from a variety of tasks, a transformer network imitating the PPD on these sets is trained. After this pre-training phase, the network is treated as fixed. In the inference phase, a fresh training set and some test features are passed to the pre-trained network, which computes predictions for the test labels in a single forward pass.

This approach is different from usual machine learning methods. Here, one would set up a model for the relationship between label and feature, and train the model parameters on a specific data set. The main benefit of PFNs is that no training or tuning is necessary at the inference stage, and predictions are delivered in split seconds.

1.2. Empirical Findings

Empirically, Müller et al. (2022) found that PFNs can indeed approximate a given PPD and perform well on real prediction tasks. While this pilot study was limited to tiny data sets, the TabPFN model of Hollmann et al. (2022) made a leap forward to classification tasks on moderately large tabular data sets. In particular, they pre-train a network with simulated data sets of size up to $n \approx 1000$ and report state-of-the-art performance on several benchmarks. And surprisingly, the network's predictions continue to improve at the inference stage when fed data sets with more than 1000 samples. This is an example of *in-context learning (ICL)*: a pre-trained network learns from the context provided in the prompt (here: the fresh training data) without updating its parameters.

1.3. Summary

The main contribution of this work is establishing the theoretical foundation for PFNs and identifying statistical mechanisms explaining their empirical behavior.

- **Theoretical framework** (Section 2): As a preliminary step, we give precise definitions of the PPD, the statistical model behind it, and its PFN approximation.
- **When PPDs can learn** (Section 3): Since PPDs are the main motivation behind PFNs, we first ask when a PPD can learn from training data. This can be approached from the perspective of Bayesian nonparametrics (Ghosal & van der Vaart, 2017). If the prior

[1]Department of Statistics, LMU Munich, Munich, Germany [2]Munich Center for Machine Learning, Munich, Germany. Correspondence to: Thomas Nagler <t.nagler@lmu.de>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

has large enough support and does not concentrate too much away from the true hypothesis, one can guarantee that the PPD converges to a close approximation of the true predictive distribution.

- **How PFNs approximate PPDs** (Section 4): The optimal PFN approximation is characterized by a Kullback-Leibler criterion. To allow for accurate approximation of the conditional class probabilities, we need sufficiently complex PFN models and prior. Practically, a PFN is trained on simulated data sets. The larger these data sets are, the more complex the PPD we approximate. The training set size can therefore be understood as a regularizer on the expected complexity of the network. The Monte-Carlo approximation of the optimal PFN is discussed briefly, but rather uneventful and of minor importance for PFNs' inference behavior.

- **Why PFNs can learn** (Section 5): The most intriguing question is: Why can a pre-trained network still learn in the inference phase? Although we know now why a PPD does, a PFN is not a valid PPD, and it is only trained to approximate the PPD for limited training sizes. The learning phenomenon can be understood through a purely frequentistic interpretation of PFNs as untrained predictors with many hyperparameters. During pre-training, these hyperparameters are tuned to be optimal for a set of tasks defined by the 'prior'. Whether the PFN predictor can learn at inference depends on its structural properties. We show that the variance of a fixed network vanishes if its sensitivity to individual samples does, and that the network's bias can only vanish if it is sufficiently localized.

- **Insights on specific PFNs** (Section 6): We look at some specific PFN models: window smoothers, classification trees, and transformer networks. The examples cover cases where the bias is constant, increasing, or decreasing with n. We show that if the model is well-designed, it can implicitly select or average over sub-models, making the bias decrease with the sample size. Transformer networks allow for vanishing variance and model selection through multi-head attention, but not for localization. However, TabPFN's bias can be improved further with a simple post-hoc localization method.

Section 7 concludes with suggestions for future research. All proofs are given in Appendix A.

1.4. Related Work

In-context-learning in large language models The recent interest in ICL was spurred by the success of *large language models (LLMs)*. These models are pre-trained on a sequence prediction task on a large corpus of text. When deployed, large models show the ability to solve tasks that they haven't seen during pre-training (e.g., mathematical reasoning problems), only from the prompt context (Brown et al., 2020; Wei et al., 2022). In particular, no parameter updates are conducted after deployment. ICL has become a new paradigm for natural language processing and is intimately linked to the transformer architecture (Vaswani et al., 2017). Dong et al. (2023) provide an up-to-date survey of the large body of LLM-related research on in-context learning.

In-context-learning on numeric data ICL has also been observed in more classical statistical learning tasks: classification and regression from tabular data. Müller et al. (2022) proposed the concept of PFNs and illustrated the abilities of a transformer model on toy examples. The TabPFN model of Hollmann et al. (2022) implements a matured version of this idea and shows superb performance on benchmarks with small tabular data. Concurrently, Nguyen & Grover (2022) proposed Transformer Neural Processes following essentially the same idea. They also consider non-*iid* settings and show promising results in applications to image completion, contextual bandits, and Bayesian Optimization. This paper illuminates the statistical foundations of such models in the *iid*-setting and disentangles the prior from the model architecture.

Mechanics of transformer-based ICL Garg et al. (2022) show that transformers can learn target functions generated from linear models, two-layer neural networks, and decision trees. Several other works provide arguments and experiments on how in-context learning emerges through implicit gradient descent (Dai et al., 2022; von Oswald et al., 2022; Akyürek et al., 2023). Olsson et al. (2022) identify a pattern of several attention heads working together, closely related to the discussion after Theorem 6.3 in this paper. Kirsch et al. (2022) experimentally investigates other architectural features (layers, memory, etc.). This work sheds further light on the mechanisms and architectural features enabling ICL.

Overall, the current work complements the existing ICL literature, by providing a theoretical foundation for the empirical findings and deriving new insights from the perspective of statistical generalization theory.

2. Theoretical Framework

2.1. Statistical Model

Consider a classification problem with class label $Y \in \mathcal{Y}$ and features $\boldsymbol{X} \in \mathcal{X} \subseteq \mathbb{R}^d$. Suppose we have *iid* training data $\mathcal{D}_n = (Y_i, \boldsymbol{X}_i)_{i=1}^n$ from some distribution p_0. The goal is to predict the conditional class probabilities $p_0(y \mid \boldsymbol{x}) =$

$\mathbb{P}(Y = y \mid \boldsymbol{X} = \boldsymbol{x})$. From the perspective of Bayesian nonparametrics, we view p_0 as a realization of a random, infinite-dimensional parameter $p \in \mathcal{P}$ with distribution Π. The distribution Π is called *prior* and expresses our beliefs about p before seeing any data. Under this model, data sets $\mathcal{D}_n \cup (Y, \boldsymbol{X})$ are generated by the following mechanism:

1. Draw $p \sim \Pi$.
2. Draw *iid* samples $\mathcal{D}_n = (Y_i, \boldsymbol{X}_i)_{i=1}^n$ and $(Y, \boldsymbol{X})$ from model p.

2.2. Posterior Predictive Distribution

For every n, the statistical model gives the tuple $(\mathcal{D}_n \cup (Y, \boldsymbol{X}), p)$ a well-defined joint distribution. For every n, we can then approximate $p_0(y \mid \boldsymbol{x})$ by the posterior predictive distribution (PPD)

$$\pi(y \mid \boldsymbol{x}, \mathcal{D}_n) = \mathbb{P}(Y = y \mid \boldsymbol{X} = \boldsymbol{x}, \mathcal{D}_n).$$

This defines a family of PPDs indexed by n. If the prior Π factorizes into independent parts for $p(y \mid \boldsymbol{x})$ and $p(\boldsymbol{x})$, the PPD can be written as

$$\pi(y \mid \boldsymbol{x}, \mathcal{D}_n) = \int p(y \mid \boldsymbol{x}) d\Pi(p \mid \mathcal{D}_n), \tag{1}$$

where the *posterior* $\Pi(\cdot \mid \mathcal{D}_n)$ is the conditional distribution of p given the data $\mathcal{D}_n$. The PPD $\pi(y \mid \boldsymbol{x}, \mathcal{D}_n)$ is then simply the posterior mean over conditional distributions $p(y \mid \boldsymbol{x})$.

Remark 2.1. In their implementation of PFNs, Müller et al. (2022) and Hollmann et al. (2022) use priors that factorize as above, but do not mention it explicitly to justify (1). Priors that do not factorize this way would lead to a different form of $\pi(y \mid \boldsymbol{x}, \mathcal{D})$:

$$\pi(y \mid \boldsymbol{x}, \mathcal{D}_n) = \int p(y \mid \boldsymbol{x}) d\Pi(p \mid \boldsymbol{x}, \mathcal{D}_n),$$

Here, observing the test feature $\boldsymbol{x}$ would be informative about the conditional distribution $p(y \mid \boldsymbol{x})$, which is unintuitive.

2.3. PFNs

A PFN is a numerical approximation of the family of PPDs. It is based on the insight that, for all n, the PPDs maximize the expected conditional likelihood

$$\mathbb{E}_\Pi[\log q(Y \mid \boldsymbol{X}, \mathcal{D}_n)], \tag{2}$$

where $\mathbb{E}_\Pi$ is an expectation over $(Y, \boldsymbol{X}) \cup \mathcal{D}_n$ generated as in Section 2.1 (see, Müller et al., 2022, Section 3).

Theorem 2.2. *Let*

$$\mathcal{Q} = \left\{ q \colon (\mathcal{Y} \times \mathcal{X})_{i=1}^{n+1} \to [0,1], \sum_{y \in \mathcal{Y}} q(y \mid \cdot, \cdot) = 1 \right\},$$

denote the set of all conditional probability functions. Then π *in* (1) *satisfies*

$$\pi = \arg\max_{q \in \mathcal{Q}} \mathbb{E}_\Pi[\log q(Y \mid \boldsymbol{X}, \mathcal{D}_n)].$$

Remark 2.3. Maximizing (2) can also be interpreted as minimizing expected KL divergence between $q(\cdot \mid \boldsymbol{X}, \mathcal{D}_n)$ and $\pi(\cdot \mid \boldsymbol{X}, \mathcal{D}_n)$.

To approximate the PPDs, we train a model $q_{\boldsymbol{\theta}}$ parametrized by $\boldsymbol{\theta}$. To be precise, for every parameter value $\boldsymbol{\theta}$, there is an entire family of functions

$$\{q_{\boldsymbol{\theta},n} \colon (\mathcal{Y} \times \mathcal{X})_{i=1}^{n+1} \to [0,1], n \in \mathbb{N}\},$$

but we shall not make this explicit in notation. To find the best parameters for given PPDs, Müller et al. (2022) propose to solve

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{\Pi_N} \mathbb{E}_\Pi[\log q_{\boldsymbol{\theta}}(Y \mid \boldsymbol{X}, \mathcal{D}_N)], \tag{3}$$

where Π_N is a probability distribution over the sample size N. The expectation over N makes $q_{\boldsymbol{\theta}^*}$ mimic the *family* of PPDs, not just its nth element.

The model $q_{\boldsymbol{\theta}}$ will normally be misspecified; that is, there is no parameter $\boldsymbol{\theta}$ such that $q_{\boldsymbol{\theta}}$ equals π. In this case, (3) defines a KL-optimal approximation of π over the class $\{q_{\boldsymbol{\theta}} \colon \boldsymbol{\theta} \in \Theta\}$. In practice, the expectation in (3) is approximated by Monte-Carlo integration, i.e., averaging over *iid* data sets $(Y_j, \boldsymbol{X}_j) \cup \mathcal{D}^{(j)}$ of size $N_j + 1$ generated as in Section 2.1 and $N_j \sim \Pi_N$. We approximate $\boldsymbol{\theta}^*$ by solving

$$\widehat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \sum_{j=1}^{m} \log q_{\boldsymbol{\theta}}(Y_j \mid \boldsymbol{X}_j, \mathcal{D}^{(j)}). \tag{4}$$

This is of course an idealization of the training process. Sophisticated PFNs are large and usually trained in a single epoch. The maximum in (4) is never reached. This does not affect the main results of the following sections, which largely consider arbitrary $\boldsymbol{\theta}$.

Remark 2.4. Hollmann et al. (2022) use a transformer network (Vaswani et al., 2017) for $q_{\boldsymbol{\theta}}$. For such architectures, any fixed network $q_{\boldsymbol{\theta}}$ accepts an arbitrary number of feature vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n_{\text{test}}}$ and a data set $\mathcal{D}_n$ of arbitrary length. The output $q_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n_{\text{test}}}, \mathcal{D}_n)$ are n_{test} vectors of conditional class probabilities. Each vector contains predictions for the conditional class probabilities $p(\cdot \mid \boldsymbol{x}_j)$. The test size n_{test} is irrelevant in what follows, so we take $n_{\text{test}} = 1$ for simplicity.

3. When PPDs can Learn

The PPDs

$$\pi(y \mid \boldsymbol{x}, \mathcal{D}_n) = \int p(y \mid \boldsymbol{x}) d\Pi(p \mid \mathcal{D}_n)$$

are fully characterized by the prior Π. If $\mathcal{D}_n$ is a data set generated from p_0, we hope that $\Pi(p \mid \boldsymbol{x}, \mathcal{D}_n)$ concentrates around p_0 as the size of $\mathcal{D}_n$ increases. Setting a good prior is tricky in a nonparametric context. Finding a prior supporting a large enough subset of possible functions isn't trivial. And even if, the prior may wash out very slowly or not at all if it puts too much mass in unfavorable regions (see, Ghosal & van der Vaart, 2017, Sections 1.2–1.3). But also if p_0 is outside the support $\mathcal{P} = \{p \colon \Pi(p) > 0\}$ of Π, PPDs can learn from data if the prior is sufficiently well-behaved:

Theorem 3.1. *Under conditions (A1) and (A2), there is $p^* \in \mathcal{P}$ such that*

$$\pi(y \mid \boldsymbol{x}, \mathcal{D}_n) \overset{n\to\infty}{\to} p^*(y \mid \boldsymbol{x}) \quad \textit{almost surely},$$

for P_0-almost every $(y, \boldsymbol{x})$. Moreover, p^* is a KL-optimal approximation of p_0 in $\mathcal{P}$.

Exact conditions and a proof are given in Appendix A.2. If $\mathcal{P}$ is sufficiently large, the KL-optimal $p^* \in \mathcal{P}$ is close to p_0. This explains why PPDs can learn when fed more data. If this was not the case, trying to approximate them with PFNs would be pointless. And the better we choose Π, the more attractive the PPDs become as an approximation target.

4. PFN Approximation of the PPD

Four factors influence the PFN approximation (4): the data prior Π, the size prior Π_N, the model $q_{\boldsymbol{\theta}}$, and the Monte-Carlo size m. Since a PFN is pre-trained, the model class $\{q_{\boldsymbol{\theta}} \colon \boldsymbol{\theta} \in \Theta\}$ can be considered fixed relative to the number of Monte-Carlo sets m. The approximation quality of $\widehat{\boldsymbol{\theta}}$ then follows from standard results on empirical risk minimization. In particular, we can expect $\widehat{\boldsymbol{\theta}} = \boldsymbol{\theta}^* + O_p(m^{-1/2})$, see Appendix B for more details. The other factors are more interesting.

If Π consists of only simple models, the optimal PFN $q_{\boldsymbol{\theta}^*}$ likely also produces only simple functions of $(y, \boldsymbol{x})$. Conversely, simple models $\{q_{\boldsymbol{\theta}} \colon \boldsymbol{\theta} \in \Theta\}$ cannot benefit much from complex Π. For the pre-trained PFN to work well on diverse tasks, we need sufficient capacity in both $\{q_{\boldsymbol{\theta}} \colon \boldsymbol{\theta} \in \Theta\}$ and Π.

When pre-training the PFN via (4), we sample data sets $\mathcal{D}^{(j)}$ with random sample size N_j. Let us define the KL-optimal parameter $\boldsymbol{\theta}_n^*$ for a given sample size:

$$\boldsymbol{\theta}_n^* = \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{\Pi}[\log q_{\boldsymbol{\theta}}(Y \mid \boldsymbol{X}, \mathcal{D}_n)].$$

The PPD $\pi(y \mid \boldsymbol{x}, \mathcal{D}_n)$ we are trying to approximate changes with n. Hence, the KL-optimal parameter $\boldsymbol{\theta}_n^*$ may change with n as well. Seen as a function of $(y, \boldsymbol{x})$, we should expect the complexity of $\pi(y \mid \boldsymbol{x}, \mathcal{D}_n)$ to increase in n. Similarly, we should expect the parameter $\boldsymbol{\theta}_n^*$ to favor more complex models. At the other extreme, $n = 1$, the true PPD is close to the average model in our prior and normally close to constant. The training set sizes N_j can thus be seen as a regularizer on model complexity. By optimizing an average over random sizes N_j, $\boldsymbol{\theta}_*$ also averages $\boldsymbol{\theta}_{N_j}^*$. The distribution Π_N lets us emphasize some ranges of sample sizes more than others. The TabPFN of Hollmann et al. (2022) was trained with a uniform distribution over $\{1, \ldots, 1023\}$ for Π_N. The restriction to small sample sizes has computational reasons: the cost of evaluating a transformer network scales quadratically in N_j.

Since TabPFN has never seen sample sizes larger than around 1000 during pre-training, it is curious that it improves its predictions when fed larger data sets. Whether such behavior occurs depends in a non-obvious way on the family $\{q_{\widehat{\boldsymbol{\theta}}}(\cdot \mid \cdot, \mathcal{D}_n), n \in \mathbb{N}\}$. The family learned by TabPFN seems to have some structure that allows extrapolating nicely to larger n. This structure may come from the architecture of the network $q_{\boldsymbol{\theta}}$ or from learning $\boldsymbol{\theta}^*$ for a given Π. The following section looks closer into the mechanisms at play.

5. Why PFNs can Learn In-Context

There is no reason to believe the PFN behaves like a PPD family for some (implicit) prior when encountering sample sizes never seen in pre-training. So even though PPDs serve as a theoretical motivation for PFNs, Theorem 3.1 does not apply to $q_{\widehat{\boldsymbol{\theta}}}$. So why does a PFN $q_{\widehat{\boldsymbol{\theta}}}$ pre-trained on up to 1000 samples improve when feeding larger data sets during inference?

5.1. PFNs as Untrained Predictors

To understand what is going on, we take a frequentist perspective. For any data size n encountered at inference, we may treat the pre-tuned network $q_{\widehat{\boldsymbol{\theta}}}(y \mid \boldsymbol{x}, \cdot)$ as an untrained predictor for $p_0(y \mid \boldsymbol{x})$, i.e., a function $(\mathcal{Y} \times \mathcal{X})^n \to \mathcal{P}_{Y|\boldsymbol{X}}$ that maps a data set $\mathcal{D}_n$ to an element of the space $\mathcal{P}_{Y|\boldsymbol{X}}$ of conditional distribution functions. In that view, $\boldsymbol{\theta}$ is a collection of tuning parameters of the predictor, selected through meta-learning in the pre-training phase. Further, the 'priors' Π and Π_N are simply distributions over tasks for which we want the predictor to do well.

Now decompose the estimation error into bias and variance components:

$$\begin{aligned} & q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - p_0(y \mid \boldsymbol{x}) \\ = \; & \underbrace{q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - \mathbb{E}_{\mathcal{D}_n \sim p_0^n}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)]}_{\text{variance}} \\ & + \underbrace{\mathbb{E}_{\mathcal{D}_n \sim p_0^n}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)] - p_0(y \mid \boldsymbol{x})}_{\text{bias}}. \end{aligned}$$

Empirically, the error above decreases with n. So what structural features of PFNs can explain this?

5.2. Symmetry

Standard transformers are symmetric functions of the individual samples in $\mathcal{D}_n$. If the samples in $\mathcal{D}_n$ are *iid*, this is most natural.

Lemma 5.1. *Let $f\colon (\mathcal{Y}\times\mathcal{X})^n \to \mathcal{P}_{Y|\boldsymbol{X}}$ be any predictor. Then there is a symmetrized version $\widetilde{f}$ of f such that, for every probability measure P,*

$$\mathbb{E}_{\mathcal{D}_n\sim P^n}[\widetilde{f}(\mathcal{D}_n)] = \mathbb{E}_{\mathcal{D}_n\sim P^n} P[f(\mathcal{D}_n)],$$
$$\text{and}\quad \mathbb{V}\mathrm{ar}_{\mathcal{D}_n\sim P^n}[\widetilde{f}(\mathcal{D}_n)] \le \mathbb{V}\mathrm{ar}_{\mathcal{D}_n\sim P^n}[f(\mathcal{D}_n)].$$

Thus, using symmetric f is optimal in an MSE sense. However, symmetry itself does not have any meaningful consequences for learning. For example, $q(y \mid \boldsymbol{x}, \mathcal{D}_n) = 1/|\mathcal{Y}|$ is a symmetric function that is incapable of learning anything.

5.3. Variance and Diminishing Sensitivity

There is other structure we can reasonably expect from $q_{\boldsymbol{\theta}}$. When passed larger data sets $\mathcal{D}_n$, the influence of individual elements should diminish. This allows to bound the variance of the predictor $q_{\boldsymbol{\theta}}$. Formally, suppose there are $\alpha > 0$ and $L < \infty$, such that for large enough n and almost all data sets $\mathcal{D}_n, \mathcal{D}_n'$ differing only in one sample,

$$\left|q_{\boldsymbol{\theta}}\big(y \mid \boldsymbol{x}, \mathcal{D}_n\big) - q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n')\right| \le L n^{-\alpha}. \tag{5}$$

Theorem 5.2. *If (5) holds, then*

$$\left|q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - \mathbb{E}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)]\right| \lesssim n^{1/2-\alpha}$$

with high probability.

If $\alpha > 1/2$, we get $\lim_{n\to\infty} n^{1/2-\alpha} = 0$, so the variance caused by $\mathcal{D}_n$ vanishes. In that case, the difference above vanishes almost surely.

Lemma 5.3. *If (5) holds with $\alpha > 1/2$, then*

$$q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - \mathbb{E}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)] \overset{n\to\infty}{\to} 0 \quad \textit{almost surely}.$$

This only partially explains how pre-trained PFNs can still learn at inference. The remaining error is due to bias.

5.4. Bias and the Need for Locality

The bias is determined by the behavior of the sequence $\mathbb{E}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)]$. It is reasonable to assume that

$$\mathbb{E}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)] \overset{n\to\infty}{\to} \overline{q}_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}),$$

for some function $\overline{q}_{\boldsymbol{\theta}}$. Without a specific model $q_{\boldsymbol{\theta}}$ at hand, we cannot say much more. In Section 6 we shall see examples where the bias is constant, and other examples where the bias decreases or increases with n.

We can give necessary conditions for a vanishing bias, however. A predictor that has vanishing bias on a sufficiently rich class of functions must be *local*: asymptotically, only samples $(Y_i, \boldsymbol{X}_i) \in \mathcal{D}_n$ with $\boldsymbol{X}_i$ close to $\boldsymbol{x}$ should contribute to $q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)$.

Theorem 5.4. *Let $\mathcal{P}$ be a set of distributions. Suppose that for every $p \in \mathcal{P}$,*

$$\mathbb{E}_{\mathcal{D}_n\sim p^n}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)] \overset{n\to\infty}{\to} p(y \mid \boldsymbol{x}). \tag{6}$$

If (5) holds, there is a sequence $\epsilon_n \to 0$ for every $\widetilde{p} \in \mathcal{P}$, such that almost surely,

$$\left|q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \widetilde{\mathcal{D}}_n)\right| \overset{n\to\infty}{\to} 0, \tag{7}$$

where $\mathcal{D}_n = (Y_i, \boldsymbol{X}_i)_{i=1}^n$ and $\widetilde{\mathcal{D}}_n = (Y_i', \boldsymbol{X}_i)_{i=1}^n$ with

$$Y_i' \begin{cases} = Y_i, & \textit{if } \|\boldsymbol{X}_i - \boldsymbol{x}\| \le \epsilon_n, \\ \sim \widetilde{p}(\cdot \mid \boldsymbol{X}_i), & \textit{if } \|\boldsymbol{X}_i - \boldsymbol{x}\| > \epsilon_n. \end{cases}$$

So if $q_{\boldsymbol{\theta}}$ is unbiased for rich enough $\mathcal{P}$, we can flip the labels of samples away from $\boldsymbol{x}$ almost arbitrarily without changing the behavior of $q_{\boldsymbol{\theta}}$; only samples $(Y_i, \boldsymbol{X}_i)$ with $\boldsymbol{X}_i$ close to $\boldsymbol{x}$ matter.

The result bears little meaning if the class $\mathcal{P}$ is too small, and meaningless if $\mathcal{P}$ contains only one p. Even for rich $\mathcal{P}$, it only provides necessary conditions for a vanishing bias. A constant predictor $q_{\boldsymbol{\theta}} = 1$ is local in the sense of (7), but its bias does not change with n. However, if $\mathcal{P}$ is rich and the bias does vanish for all $p \in \mathcal{P}$, the predictor $q_{\boldsymbol{\theta}}$ can effectively only use $\epsilon_n n = o(n)$ samples, so (5) is unlikely to hold with $\alpha = 1$. This is in line with the lower bounds on the bias-variance trade-off derived in Derumigny & Schmidt-Hieber (2020).

6. Insights on Specific PFNs

We now consider some concrete examples of PFNs $q_{\boldsymbol{\theta}}$, to shed further light on the factors facilitating learning in the inference phase. Before turning to transformer networks, we briefly discuss two simpler models to illustrate some key mechanisms. The following result will be helpful.

Lemma 6.1. *Let g be a function bounded by $K < \infty$ and*

$$q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) = \frac{\sum_{i=1}^n g(y, \boldsymbol{x}, Y_i, \boldsymbol{X}_i)}{\sum_{i=1}^n \mathbb{1}\{\boldsymbol{X}_i \in A_n(\boldsymbol{x})\}},$$

for some sequence $A_n(\boldsymbol{x}) \subset \mathcal{X}$. If

$$n^\eta \mathbb{P}\{\boldsymbol{X}_i \in A_n(\boldsymbol{x})\} \overset{n\to\infty}{\to} c,$$

for some $\eta \in (0, 1/2)$ and $c > 0$, then $q_{\boldsymbol{\theta}}$ satisfies the conditions of Theorem 5.2 with $\alpha = 1 - \eta$ and $L = 4K/c$.

6.1. Window Smoother

For $\theta \in (0, \infty)$, define

$$q_\theta(y \mid \boldsymbol{x}, \mathcal{D}_n) = \frac{\sum_{i=1}^n \mathbb{1}(Y_i = y)\mathbb{1}(\|\boldsymbol{X}_i - \boldsymbol{x}\| < \theta)}{\sum_{i=1}^n \mathbb{1}(\|\boldsymbol{X}_i - \boldsymbol{x}\| < \theta)}.$$

This corresponds to a window smoother with bandwidth θ. Then θ^* is the KL-optimal bandwidth for datasets from the prior. The fitted PFN is therefore just a window smoother with its hyperparameter tuned to such data sets. According to Lemma 6.1, q_θ satisfies (5) with $\alpha = 1$. The bias

$$\begin{aligned}&\mathbb{E}[q_\theta(y \mid \boldsymbol{x}, \mathcal{D}_n)] - p_0(y \mid \boldsymbol{x})\\ &= \mathbb{P}_0(Y = y \mid \|\boldsymbol{X} - \boldsymbol{x}\| < \theta) - p_0(y \mid \boldsymbol{x})\end{aligned}$$

is constant, but optimized for data sizes from $\Pi \times \Pi_N$. Despite constant bias, the PFN learns from more data at inference, but only through reducing its variance.

Now consider some sequence $(a_n)_{n\in\mathbb{N}}$ and

$$q_\theta(y \mid \boldsymbol{x}, \mathcal{D}_n) = \frac{\sum_{i=1}^n \mathbb{1}(Y_i = y)\mathbb{1}(\|\boldsymbol{X}_i - \boldsymbol{x}\| < a_n\theta)}{\sum_{i=1}^n \mathbb{1}(\|\boldsymbol{X}_i - \boldsymbol{x}\| < a_n\theta)}.$$

If a_n increases with n, the width of the smoothing window does too. This choice of q_θ isn't sensible, of course, as the bias

$$\mathbb{P}_0(Y = y \mid \|\boldsymbol{X} - \boldsymbol{x}\| < a_n\theta) - p_0(y \mid \boldsymbol{x})$$

typically increases with n. If we instead choose $a_n \to 0$ and p_0 is sufficiently smooth, we can get rid of the bias. The scaling $a_n = n^{-1/(4+d)}$ is known to be optimal in an MSE sense (e.g., Wand & Jones, 1994), with squared bias and variance decreasing at rate $n^{-4/(4+d)}$. Indeed, we have $a_n\mathbb{P}_0(\|\boldsymbol{X} - \boldsymbol{x}\| < a_n\theta) \to p_0(\boldsymbol{x})$, so this is the convergence rate implied by Lemma 6.1 and Theorem 5.2. The hyperparameter θ^* reduces to a prefactor, tuned to be (asymptotically) optimal for data sets generated from Π of arbitrary size.

6.2. Classification Trees

To keep the notation simple, suppose for the moment that $\mathcal{X} \subseteq \mathbb{R}$. Define

$$\begin{aligned}&q_{\boldsymbol{\theta}}(y \mid x, \mathcal{D})\\ &= \sum_{j=0}^S \mathbb{1}_{(\theta_j,\theta_{j+1}]}(x) \frac{\sum_{i=1}^n \mathbb{1}(Y_i = y)\mathbb{1}_{(\theta_j,\theta_{j+1}]}(X_i)}{\sum_{i=1}^n \mathbb{1}_{(\theta_j,\theta_{j+1}]}(X_i)},\end{aligned}$$

as a classification tree with parameters $\boldsymbol{\theta} \in \mathbb{R}^S$ and $\theta_0 = -\infty, \theta_{S+1} = +\infty$ by convention. The (hyper-)parameters are the split locations of the tree. The split locations are trained offline, to work best on sets from the prior $\Pi \times \Pi_N$. Lemma 6.1 yields that Theorem 5.2 holds with $\alpha = 1$. The bias is

$$\sum_{j=1}^S \mathbb{1}_{(\theta_j,\theta_{j+1}]}(x)\mathbb{P}_0(Y = y \mid X \in (\theta_j, \theta_{j+1}]) - p_0(y \mid x),$$

which is independent of n. To reduce the bias as in the previous example, we would need to grow the number S of split locations with n. But the model is considered fixed in the inference phase and we cannot change the number of parameters.

Instead, we could set up an ensemble of classification trees with Bayesian model averaging. Let $q_{\boldsymbol{\theta}_1}, \ldots, q_{\boldsymbol{\theta}_K}$ be classification trees as above, and

$$q_{\boldsymbol{\theta}}(y \mid x, \mathcal{D}_n) = \frac{1}{K}\sum_{k=1}^K q_{\boldsymbol{\theta}_k}(y \mid x, \mathcal{D}_n) w(\mathcal{D}_n; \boldsymbol{\theta}_k),$$

where

$$w(\mathcal{D}_n; \boldsymbol{\theta}_k) = \frac{\exp\{-\mathrm{BIC}(\mathcal{D}_n; \boldsymbol{\theta}_k)\}}{\sum_{j=1}^K \exp\{-\mathrm{BIC}(\mathcal{D}_n; \boldsymbol{\theta}_j)\}},$$

and

$$\mathrm{BIC}(\mathcal{D}_n; \boldsymbol{\theta}_k) = -2\sum_{i=1}^n \log q_{\boldsymbol{\theta}_k}(Y_i \mid X_i, \mathcal{D}_n) + S\log n.$$

As n grows, the model $q_{\boldsymbol{\theta}}$ drifts towards a weighted average of the best-performing ensemble members. We can thus expect the bias to reduce with n, approaching the bias of the best ensemble members. The role of the hyperparameters $\boldsymbol{\theta}$ is the same as for a single tree. But now, the KL-optimal parameter $\boldsymbol{\theta}^*$ likely induces more complex and diverse ensemble members.

6.3. Transformer Networks

We now consider a transformer network with one layer. Let $\mathcal{D}_n = \{\boldsymbol{V}_i\}_{i=1}^n$ with $\boldsymbol{V}_i = (Y_i, \boldsymbol{X}_i) \in \{0, 1\} \times \mathbb{R}^d$ and $\boldsymbol{v} = (0, \boldsymbol{x})$. Similar[1] to Thickstun (2021), define

$$\begin{aligned}a_j^{(h)} &= \mathrm{SoftMax}\big(\boldsymbol{v}^\top W_q^{(h)}\boldsymbol{V}_1, \ldots, \boldsymbol{v}^\top W_q^{(h)}\boldsymbol{V}_n\big)_j,\\ \boldsymbol{u}' &= \sum_{h=1}^H\sum_{j=1}^n a_j^{(h)} W_v^{(h)}\boldsymbol{V}_j,\\ \boldsymbol{u} &= \mathrm{LayerNorm}(\boldsymbol{v} + \boldsymbol{u}'; \boldsymbol{\gamma}),\\ \boldsymbol{z}' &= W_{r,2}\,\mathrm{ReLu}(W_{r,1}\boldsymbol{u}; \boldsymbol{\gamma}),\\ \boldsymbol{z} &= \mathrm{LayerNorm}(\boldsymbol{u} + \boldsymbol{z}'; \boldsymbol{\gamma}),\\ q_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}, \mathcal{D}_n) &= \mathrm{SoftMax}(W_o\boldsymbol{z}),\end{aligned}$$

[1] Some scaling and redundancies in the parametrization have been deliberately removed. They help for training the network, but not its theoretical analysis.

where $W_q^{(h)}, W_v^{(h)} \in \mathbb{R}^{(d+1)\times(d+1)}$, $W_{r,1}, W_{r,2}^\top \in \mathbb{R}^{m\times(d+1)}$, and $W_o \in \mathbb{R}^{|\mathcal{Y}|\times(d+1)}$. The parameter $\boldsymbol{\theta}$ collects all these matrices. The SoftMax, LayerNorm, and ReLu operations are defined as

$$\begin{aligned}
\mathrm{SoftMax}(\boldsymbol{v}) &= \frac{\exp(\boldsymbol{v})}{\sum_{j=1}^d \exp(v_j)}, \\
\mathrm{LayerNorm}(\boldsymbol{v};\boldsymbol{\gamma}) &= \gamma_1 \frac{\boldsymbol{v} - \mathrm{avg}(\boldsymbol{v})}{\|\boldsymbol{v} - \mathrm{avg}(\boldsymbol{v})\| + |\gamma_2|} + \gamma_3, \\
\mathrm{avg}(\boldsymbol{v}) &= \frac{1}{d}\sum_{j=1}^d v_j \mathbf{1}, \\
\mathrm{ReLu}(\boldsymbol{v}) &= \max(0, \boldsymbol{v}),
\end{aligned}$$

with exp and max acting componentwise on vectors. Here and in everything that follows, the norm $\|\cdot\|$ is understood as $\|\cdot\|_2$ for both vectors and matrices.

The first two equations describe an *attention mechanism* with H heads (Vaswani et al., 2017). By definition, $a_1^{(h)} + \cdots + a_n^{(h)} = 1$. The idea is that within every head, the attention weights $a_j^{(h)}$ emphasize specific samples $\boldsymbol{V}_j \in \mathcal{D}_n$. Emphasis is put on those $\boldsymbol{V}_j$ that are 'similar' to $\boldsymbol{v}$ in a sense measured by $\boldsymbol{v}^\top W_q^{(h)} \boldsymbol{V}_j$. Each attention head allows for a different definition of similarity. With the help of Theorem 5.2 and Theorem A.1, we can show that the variance of this predictor vanishes.

Theorem 6.2. *For $\mathcal{X} = \{\boldsymbol{x} \colon \|\boldsymbol{x}\| \le K\}$ and $\|W_q^{(h)}\|, \|W_v^{(h)}\|, \|W_{r,1}\|, \|W_{r,2}\|, \|W_o\| < \infty$, it holds*

$$\left|q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - \mathbb{E}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)]\right| \lesssim n^{-1/2},$$

with high probability.

The variance vanishes irrespective of the parameter $\boldsymbol{\theta}$. This is because the attention mechanism necessarily gives diminishing weight to individual samples (see the proof in Appendix A.8).

The bias depends heavily on the choice of $\boldsymbol{\theta}$.

Theorem 6.3. *Under the assumptions of Theorem 6.2,*

$$\mathbb{E}[q_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}, \mathcal{D}_n)] \overset{n\to\infty}{\to} \overline{q}_\theta(\cdot \mid \boldsymbol{x}),$$

where $\overline{q}_\theta(\cdot \mid \boldsymbol{x})$ is defined as $q_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}, \mathcal{D}_n)$, but with $\boldsymbol{u}'$ replaced by

$$\overline{\boldsymbol{u}}' = \sum_{h=1}^H W_v^{(h)} \mathbb{E}_{\boldsymbol{V}\sim g_h}[\boldsymbol{V}],$$

$$\textit{where} \quad g_h(\boldsymbol{s}) = \frac{\exp(\boldsymbol{v}^\top W_q^{(h)} \boldsymbol{s})}{\mathbb{E}_{\boldsymbol{V}\sim p_0}[\exp(\boldsymbol{v}^\top W_q^{(h)} \boldsymbol{V})]} p_0(\boldsymbol{s}).$$

The measure g_h is an exponentially tilted version of p_0 (Siegmund, 1976). The tilt can be understood as an infinitesimal form of the attention mechanism. Relative to p_0, the tilted measure lifts the likelihood of values $\boldsymbol{s}$ that are similar to $\boldsymbol{v}$ (with similarity measured by $\boldsymbol{v}^\top W_q^{(h)} \boldsymbol{s}$) and discounts the likelihood of others. Each attention head h assesses a certain aspect of the unknown distribution p_0 (characterized by $W_q^{(h)}$). If the matrices $(W_q^{(h)})_{h=1}^H$ are specified well, the aspect views distinguish distinct feature values. This localizes the predictor to some degree, but not in the sense of Theorem 5.4. Although we upweight the influence of samples "similar to" $\boldsymbol{v}$, there always remains an influence of samples away from $\boldsymbol{v}$. We cannot flip the labels of such samples without changing the predictor (asymptotically), so we should not expect the bias to vanish.

Nevertheless, the limiting bias $\overline{q}_\theta(y \mid \boldsymbol{x}) - p_0(y \mid \boldsymbol{x})$ may be small if the remaining network processes *the sum* of aspect summaries $W_v^{(h)} \mathbb{E}_{\boldsymbol{V}\sim g_h}[\boldsymbol{V}]$ into a good approximation of p_0. The relevance of individual aspect summaries depends on the true measure p_0, and less relevant aspects may also contribute less to the sum. On small samples, this effect is milder. At the extreme end, $n = 1$, all attention weights $a_j^{(h)}$ equal 1, so all aspects contribute equally. This suggests that the bias of the transformer network may decrease — provided the hyperparameters downstream make meaningful use of the aspect views. For example, Olsson et al. (2022) identified powerful patterns of several attention heads working together. This effect is similar to that of the model averaging layer in Section 6.2. Key to this is the presence of multiple attention heads ($H > 1$). However, this applies only to sample sizes the parameter $\boldsymbol{\theta}$ has been tuned to. For larger sample sizes, there is no reason to expect a tuned network's bias to decrease.

6.4. Localized PFNs

According to Theorem 5.4, we need to localize the network to make its bias decrease. A simple post-hoc approach applicable to any pre-trained network is the following. To predict the label at a new feature $\boldsymbol{x}$:

1. Construct a reduced training set $\mathcal{D}_n(\boldsymbol{x})$ by excluding all but the k_n nearest neighbors of $\boldsymbol{x}$ from $\mathcal{D}_n$.
2. Predict the label that maximizes $q_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}, \mathcal{D}_n(\boldsymbol{x}))$.

Intuitively, restricting to a neighborhood is like stretching/flattening the target $p(y \mid \cdot)$ at the cost of a reduction in sample size. Flatter functions are easier to approximate. This is the mechanism behind the window smoother from Section 6.1. If the model q_θ approximates constant functions well, localization should improve the bias.

6.5. Numerical Validation

Since the key mechanism acting on $\mathcal{D}_n$ remains intact if we add more layers to the network, the findings likely transfer to

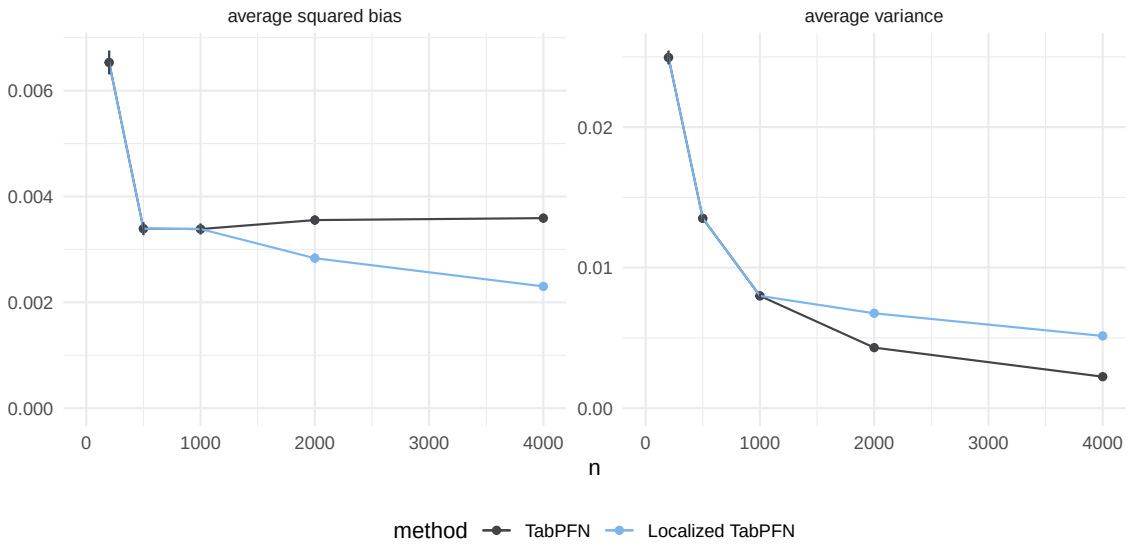

Figure 1. Average squared bias and variance of the pre-trained TabPFN of Hollmann et al. (2022) on simulated data sets.

larger networks. The main predictions from our theoretical considerations are: (i) the variance vanishes at rate $1/n$, (ii) the bias does not vanish, but decreases until $n \approx 1000$. To confirm this empirically, we simulate 500 data sets $\mathcal{D}_n$ from the model $p_0(1 \mid \boldsymbol{X}) = 1/2 + \sin(\mathbf{1}^\top \boldsymbol{X})/2$ with $Y \in \{0, 1\}$, $\boldsymbol{X} \sim \mathcal{N}(\mathbf{0}, I_5)$, and run the pre-trained TabPFN of Hollmann et al. (2022, pip version 0.1.8).[2] We compute the average squared bias and variance over 100 samples $\boldsymbol{X}_{\text{test}} \sim \mathcal{N}(\mathbf{0}, I_5)$. The results in Figure 1 confirm that the variance indeed decreases at rate $1/n$ and that the bias decreases until $n \approx 1000$, but does not vanish.

The analysis shows that, for larger sample sizes at inference, TabPFN learns mainly through decreasing its variance. This variance reduction is a consequence of the transformer architecture and takes place irrespective of the tuned parameters $\widehat{\boldsymbol{\theta}}$. Figure 1 also shows the results of a localized version of TabPFN (as in Section 6.4 with $k_n = \min\{500, \lceil n^{4/(d+4)} \rceil\}$). Here, the bias continues to decrease beyond $n = 1000$ at the cost of a slightly larger variance.

7. Discussion

As explained in Section 5.1, the prior Π characterizes tasks we want the predictor to do well on. Hollmann et al. (2022) propose a new kind of prior based on structural causal models (Pearl, 2009) that is interesting on its own. Their intuitive idea is that the pair $(Y, \boldsymbol{X})$ is generated by some noisy, causal mechanism (not necessarily in the direction $\boldsymbol{X} \to Y$). Because the mechanisms can be arbitrarily complex, the prior is essentially nonparametric. The rate at which corresponding posteriors contract is a complex issue (Ghosal & van der Vaart, 2017, Chapters 8–9) and poses an interesting open question.

The key factors driving PFNs capability to learn are their sensitivity to individual samples, their ability to choose submodels, and localization. These insights may help to inform architecture design. In Section 6.1 and Section 6.4, we found a way to make the bias vanish at the cost of increased sensitivity to the training instances. This was achieved by introducing a scaling of the tuning parameters adapted to the training set size n. Whether this is possible and what a good scaling is depends on the model architecture. The localization approach in Section 6.4 is simple and can be applied post-hoc to any pre-trained network q_θ. More serious architecture design should account for the entire training pipeline and computational efficiency. Additional improvements can be expected if localization is incorporated into pre-training. Thinking about ways to adapt the transformer architecture appropriately could be a promising path. Another possible improvement is to augment the architecture with a Bayesian averaging mechanism similar to Section 6.2.

Hollmann et al. (2022) acknowledge constraints on the feature dimension and sample size as a major limitation of current PFN implementations. Owing to the standard transformer architecture, the maximal feature size is fixed, and the algorithm scales quadratically in the number of samples. To mitigate this, several works proposed scalable modifications of the transformer architecture (Beltagy et al., 2020;

[2] An R script to reproduce the results can be found at https://gist.github.com/tnagler/62f6ce1f996333c799c81f1aef147e72.

Zaheer et al., 2020; Kitaev et al., 2020). Hollmann et al. (2022) rightfully point out that PFNs are quick enough to be used as ensemble members. The size constraints could therefore be overcome by boosting and bagging techniques akin to random forests or boosted trees.

The full potential of PFNs is yet to be explored.

Acknowledgements

The author is grateful for several helpful comments by Samuel Müller, Noah Hollmann, Thibault Vatter, and two anonymous referees.

References

Akyürek, E., Schuurmans, D., Andreas, J., Ma, T., and Zhou, D. What learning algorithm is in-context learning? investigations with linear models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=0g0X4H8yN4I.

Beltagy, I., Peters, M. E., and Cohan, A. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Boucheron, S., Lugosi, G., and Massart, P. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press, 2013.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Dai, D., Sun, Y., Dong, L., Hao, Y., Sui, Z., and Wei, F. Why can gpt learn in-context? language models secretly perform gradient descent as meta optimizers. *arXiv preprint arXiv:2212.10559*, 2022.

De Blasi, P. and Walker, S. G. Bayesian asymptotics with misspecified models. *Statistica Sinica*, pp. 169–187, 2013.

Derumigny, A. and Schmidt-Hieber, J. On lower bounds for the bias-variance trade-off. *arXiv preprint arXiv:2006.00278*, 2020.

Dong, Q., Li, L., Dai, D., Zheng, C., Wu, Z., Chang, B., Sun, X., Xu, J., Li, L., and Sui, Z. A survey on in-context learning, 2023.

Gao, B. and Pavel, L. On the properties of the softmax function with application in game theory and reinforcement learning. *arXiv preprint arXiv:1704.00805*, 2017.

Garg, S., Tsipras, D., Liang, P. S., and Valiant, G. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598, 2022.

Ghosal, S. and van der Vaart, A. *Fundamentals of nonparametric Bayesian inference*, volume 44. Cambridge University Press, 2017.

Hollmann, N., Müller, S., Eggensperger, K., and Hutter, F. TabPFN: A transformer that solves small tabular classification problems in a second. In *NeurIPS 2022 First Table Representation Workshop*, 2022. URL https://openreview.net/forum?id=eu9fVjVasr4.

Kirsch, L., Harrison, J., Sohl-Dickstein, J., and Metz, L. General-purpose in-context learning by meta-learning transformers, 2022.

Kitaev, N., Kaiser, L., and Levskaya, A. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

McDiarmid, C. *On the method of bounded differences*, pp. 148–188. London Mathematical Society Lecture Note Series. Cambridge University Press, 1989. doi: 10.1017/CBO9781107359949.008.

Müller, S., Hollmann, N., Arango, S. P., Grabocka, J., and Hutter, F. Transformers can do bayesian inference. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=KSugKcbNf9.

Nguyen, T. and Grover, A. Transformer neural processes: Uncertainty-aware meta learning via sequence modeling. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 16569–16594. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/nguyen22b.html.

Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen,

A., Conerly, T., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Johnston, S., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

Pearl, J. *Causality*. Cambridge university press, 2009.

Siegmund, D. Importance Sampling in the Monte Carlo Study of Sequential Tests. *The Annals of Statistics*, 4(4):673 – 684, 1976. doi: 10.1214/aos/1176343541. URL https://doi.org/10.1214/aos/1176343541.

Thickstun, J. The transformer model in equations. *University of Washington, Tech. Rep*, 2021. URL https://johnthickstun.com/docs/transformers.pdf.

Vaart, A. W. and Wellner, J. A. Weak convergence. In *Weak convergence and empirical processes*, pp. 16–28. Springer, 1996.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

von Oswald, J., Niklasson, E., Randazzo, E., Sacramento, J., Mordvintsev, A., Zhmoginov, A., and Vladymyrov, M. Transformers learn in-context by gradient descent. *arXiv preprint arXiv:2212.07677*, 2022.

Wand, M. P. and Jones, M. C. *Kernel smoothing*. CRC press, 1994.

Wei, J., Bosma, M., Zhao, V., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=gEZrGCozdqR.

Zaheer, M., Guruganesh, G., Dubey, K. A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q., Yang, L., et al. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297, 2020.

A. Proofs

A.1. Proof of Theorem 2.2

By definition of the KL-divergence,

$$0 \leq \mathrm{KL}\big[q(\cdot \mid \boldsymbol{x}, \mathcal{D}) \mid\mid \pi(\cdot \mid \boldsymbol{x}, \mathcal{D})\big] \quad \text{for all } (\boldsymbol{x}, \mathcal{D}),$$

or, equivalently,

$$\mathbb{E}_{Y \sim \pi(\cdot \mid \boldsymbol{x}, \mathcal{D})}[\log \pi(Y \mid \boldsymbol{x}, \mathcal{D})] \geq \mathbb{E}_{Y \sim \pi(\cdot \mid \boldsymbol{x}, \mathcal{D})}[\log q(Y \mid \boldsymbol{x}, \mathcal{D})] \quad \text{for all } (\boldsymbol{x}, \mathcal{D}).$$

Since this holds for any $(\boldsymbol{x}, \mathcal{D})$ it must also hold if we take expectations over random draws of $(\boldsymbol{x}, \mathcal{D})$. Taking expectation with respect to $(\boldsymbol{x}, \mathcal{D}) \sim \Pi$ on both sides, the law of iterated expectations yields

$$\mathbb{E}_{\Pi}[\log \pi(Y \mid \boldsymbol{X}, \mathcal{D})] \geq \mathbb{E}_{\Pi}[\log q(Y \mid \boldsymbol{X}, \mathcal{D})]. \qquad \square$$

A.2. Proof of Theorem 3.1

Write $\mathrm{KL}(f \mid f')$ for the KL-divergence of f relative to f', and $H(f, f')$ for their Hellinger distance. We need the following assumptions:

(A1) There is a unique $p^* \in \mathcal{P}$ with $p^* = \arg\min_p \mathrm{KL}(p^* \mid p_0)$ and $\mathrm{KL}(p^* \mid p_0) < \infty$.

(A2) For every $\alpha \in (0, 1/2)$, there are sets $B_1, \ldots, B_{J(\alpha)}$ with

$$\mathcal{P} \subseteq \bigcup_{j=1}^{J(\alpha)} B_j, \qquad \sup_{p, p' \in B_j} H(p, p') \leq 4(\alpha^2/2)^{1/\alpha}, \qquad \sum_{j=1}^{J(\epsilon)} \Pi(B_j)^{\alpha} < \infty.$$

Now let $\Pi_n(A) = \int_A d\Pi(p \mid \mathcal{D}_n)$ be the posterior measure. From our assumptions and Corollary 1 of De Blasi & Walker (2013) it follows that for all $\epsilon > 0$,

$$\Pi_n\big\{p \in \mathcal{P} \colon H(p, p^*) > \epsilon\big\} \to 0, \tag{8}$$

with probability 1 over sequences $\mathcal{D}_n$. For some $\delta > 0$ and an arbitrary set $A \subseteq \mathcal{Y} \times \mathcal{X}$ with $\mu_A = P^*(A) > 0$, define

$$S_{\delta, A} = \Big\{p \in \mathcal{P} \colon \inf_{(y, \boldsymbol{x}) \in A} \big|p(y \mid \boldsymbol{x}) - p^*(y \mid \boldsymbol{x})\big| \geq \delta\Big\}.$$

For any $p \in S_{\delta, A}$, it holds

$$\begin{aligned}
\mu_A \delta = \int_A \delta p^*(\boldsymbol{x}) d\boldsymbol{x} &\leq \int_A |p^*(y \mid \boldsymbol{x}) - p(y \mid \boldsymbol{x})| p^*(\boldsymbol{x}) d\boldsymbol{x} \\
&= \int_A \big|p^*(y, \boldsymbol{x}) - p(y \mid \boldsymbol{x}) p^*(\boldsymbol{x})\big| d\boldsymbol{x} \\
&= \int_A \big|p^*(y, \boldsymbol{x}) - p(y, \boldsymbol{x}) + p(y \mid \boldsymbol{x})[p(\boldsymbol{x}) - p^*(\boldsymbol{x})]\big| d\boldsymbol{x} \\
&\leq \int \big|p^*(y, \boldsymbol{x}) - p(y, \boldsymbol{x})\big| d\boldsymbol{x} + \int \big|p(\boldsymbol{x}) - p^*(\boldsymbol{x})\big| d\boldsymbol{x} \\
&\leq 4\, \mathrm{TV}(p^*, p) \\
&\leq 8 H(p^*, p),
\end{aligned}$$

where $\mathrm{TV}(f, f')$ is the total variation distance. Together with (8), this implies

$$\Pi_n(S_{\delta, A}) \leq \Pi_n\big\{p \colon H(p, p^*) \geq \mu_A \delta / 8\big\} \to 0.$$

We then get

$$\begin{aligned}\inf_{(y,\boldsymbol{x})\in A} |\pi(y \mid \boldsymbol{x}, \mathcal{D}_n) - p_0(y \mid \boldsymbol{x})| &\leq \inf_{(y,\boldsymbol{x})\in A} \left| \int p(y \mid \boldsymbol{x}) d\Pi_n(p) - p_0(y \mid \boldsymbol{x}) \right| \\ &\leq \inf_{(y,\boldsymbol{x})\in A} \int \big| p(y \mid \boldsymbol{x}) - p_0(y \mid \boldsymbol{x}) \big| d\Pi_n(p) \\ &\leq \inf_{(y,\boldsymbol{x})\in A} \int_{S_{\delta,A}} \big| p(y \mid \boldsymbol{x}) - p_0(y \mid \boldsymbol{x}) \big| d\Pi_n(p) + \delta \\ &\leq 2\Pi_n(S_{\delta,A}) + \delta \to \delta \quad \text{almost surely.}\end{aligned}$$

Since δ and A were arbitrary, we have shown that

$$\pi(y \mid \boldsymbol{x}, \mathcal{D}_n) \to p*(y \mid \boldsymbol{x}),$$

with probability 1 for P^*-almost every $(y, \boldsymbol{x})$. Since $\mathrm{KL}(p^* \mid p_0) < \infty$ by (A1), convergence must also take place for P_0-almost every $(y, \boldsymbol{x})$. □

A.3. Proof of Lemma 5.1

Let R_n be the set of permutations $\rho\colon (\mathcal{Y} \times \mathcal{X})^n \to (\mathcal{Y} \times \mathcal{X})^n$. Define the symmetrized function $\widetilde{f}$ as

$$\widetilde{f}(\mathcal{D}_n) = \frac{1}{|R_n|} \sum_{\rho \in R_n} (f \circ \rho)(\mathcal{D}_n).$$

If the elements in $\mathcal{D}_n$ are *iid*, it holds $\mathbb{E}_P[f(\mathcal{D}_n)] = \mathbb{E}_P[(f \circ \rho)(\mathcal{D}_n)]$ and $\mathbb{V}\mathrm{ar}_P[f(\mathcal{D}_n)] = \mathbb{V}\mathrm{ar}_P[(f \circ \rho)(\mathcal{D}_n)]$ for any $\rho \in R_n$. Therefore, $\mathbb{E}_P[f(\mathcal{D}_n)] = \mathbb{E}_P[\widetilde{f}(\mathcal{D}_n)]$ and

$$\mathbb{V}\mathrm{ar}_P[\widetilde{f}(\mathcal{D}_n)] = \frac{\mathbb{V}\mathrm{ar}_P[f(\mathcal{D}_n)]}{|R_n|^2} \sum_{\rho,\rho' \in R_n} \mathrm{Corr}_P[(f \circ \rho)(\mathcal{D}_n), (f \circ \rho')(\mathcal{D}_n)] \leq \mathbb{V}\mathrm{ar}_P[f(\mathcal{D}_n)],$$

with equality for all P if and only if f is symmetric. □

A.4. Proof of Theorem 5.2

McDiarmid's inequality (McDiarmid, 1989) yields

$$\mathbb{P}\big(\big|q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - \mathbb{E}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)]\big| > \epsilon\big) \leq 2\exp\left(-\frac{2\epsilon^2}{L^2 n^{1-2\alpha}}\right).$$

Choosing $\epsilon^2 = \log(\delta) L^2 n^{1-2\alpha}/2$, we get

$$\mathbb{P}\big(\big|q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - \mathbb{E}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)]\big| > \log(\delta)^{1/2} K L n^{1/2-\alpha}/\sqrt{2}\big) \leq 2\delta.$$ □

A.5. Proof of Lemma 5.3

Using the first inequality from the previous proof, we get

$$\sum_{n=1}^{\infty} \mathbb{P}\big(\big|q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - \mathbb{E}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)]\big| > \epsilon\big) \leq 2\sum_{n=1}^{\infty} \exp\left(-\frac{2\epsilon^2}{L^2 n^{1-2\alpha}}\right) < \infty.$$

The Borel-Cantelli lemma then implies that, almost surely,

$$\lim_{n\to\infty} \big|q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - \mathbb{E}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)]\big| \leq \epsilon$$

Since ϵ was arbitrary the claim follows. □

A.6. Proof of Theorem 5.4

Condition (6) implies

$$\mathbb{E}_p[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)] - \mathbb{E}_{\widetilde{p}}[q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n)] \to 0,$$

for all $p, \widetilde{p} \in \mathcal{P}$ with $p(y \mid \boldsymbol{s}) = \widetilde{p}(y \mid \boldsymbol{s})$ for $\|\boldsymbol{s} - \boldsymbol{x}\| < \epsilon$. Lemma 5.3 then implies

$$\lim_{n\to\infty} \left| q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \widetilde{\mathcal{D}}_n) \right| = 0 \quad \text{almost surely.}$$

Since ϵ was arbitrary, convergence must also hold for some sequence $\epsilon_n \to 0$. □

A.7. Proof of Lemma 6.1

Let $\mathcal{D}_n = (Y_i, \boldsymbol{X}_i)_{i=1}^n$ and $\mathcal{D}'_n = (Y'_i, \boldsymbol{X}'_i)_{i=1}^n$ such that $(Y_i, \boldsymbol{X}_i) = (Y'_i, \boldsymbol{X}'_i)$ for all $i > 1$. Hoeffdings's inequality (Boucheron et al., 2013, Theorem 2.8) gives

$$\mathbb{P}\left(\left| \frac{1}{n} \sum_{i=1}^n \mathbb{1}\{\boldsymbol{X}_i \in A_n(\boldsymbol{x})\} - \mathbb{P}\{\boldsymbol{X}_i \in A_n(\boldsymbol{x})\} \right| > cn^{-\eta}/3 \right) \leq 2\exp\bigl(-c^2 n^{1-2\eta}/9\bigr).$$

Since $\eta < 1/2$,

$$\sum_{n=1}^\infty \exp\bigl(-c^2 n^{1-2\eta}/9\bigr) < \infty,$$

and the Borell-Cantelli lemma implies that for large n,

$$\left| \frac{1}{n} \sum_{i=1}^n \mathbb{1}\{\boldsymbol{X}_i \in A_n(\boldsymbol{x})\} - \mathbb{P}\{\boldsymbol{X}_i \in A_n(\boldsymbol{x})\} \right| \leq cn^{-\eta}/3 \quad \text{almost surely.}$$

The remaining inequalities are understood almost surely, for large enough n. Because $n^\eta \mathbb{P}\{\boldsymbol{X}_i \in A_n(\boldsymbol{x})\} \to c$, we get

$$\begin{aligned}
& n^\eta \left| \frac{1}{n} \sum_{i=1}^n \mathbb{1}\{\boldsymbol{X}_i \in A_n(\boldsymbol{x})\} - cn^{-\eta} \right| \\
& \leq n^\eta \left| \frac{1}{n} \sum_{i=1}^n \mathbb{1}\{\boldsymbol{X}_i \in A_n(\boldsymbol{x})\} - \mathbb{P}\{\boldsymbol{X}_i \in A_n(\boldsymbol{x})\} \right| + \left| n^\eta \mathbb{P}\{\boldsymbol{X}_i \in A_n(\boldsymbol{x})\} - c \right| \\
& \leq c/2,
\end{aligned}$$

which implies

$$\frac{1}{n} \sum_{i=1}^n \mathbb{1}\{\boldsymbol{X}_i \in A_n(\boldsymbol{x})\} \geq cn^{-\eta}/2.$$

Using this bound,

$$n^{1-\eta} |q_\theta(y \mid \boldsymbol{x}, \mathcal{D}_n) - q_\theta(y \mid \boldsymbol{x}, \mathcal{D}'_n)| \leq 2|g(y, \boldsymbol{x}, Y_1, \boldsymbol{X}_1) - g(y, \boldsymbol{x}, Y'_1, \boldsymbol{X}'_1)|/c \leq 4K/c,$$

which proves the claim.

A.8. Proof of Theorem 6.2

The theorem is a consequence of Theorem 5.2 and the following result. (The norm bounds on $\|\boldsymbol{x}\|$ and the weight matrices are arbitrary and can be relaxed.)

Theorem A.1. *Let $\mathcal{X} = \{\boldsymbol{x} \colon \|\boldsymbol{x}\| \leq 1\}$ and $\|W_q^{(h)}\|, \|W_v^{(h)}\|, \|W_{r,1}\|, \|W_{r,2}\|, \|W_o\| \leq 1$. Then the network $q_{\boldsymbol{\theta}}$ satisfies (5) with $\alpha = 1$ and $L = O(H|\gamma_1|/|\gamma_2|)$.*

Proof. Let $\widetilde{\mathcal{D}}_n = (\mathcal{D}_n \setminus \boldsymbol{V}_n) \cup \widetilde{\boldsymbol{V}}_n$ and define $\widetilde{a}_j^{(h)}$, $\widetilde{\boldsymbol{u}}'$, $\widetilde{\boldsymbol{u}}$, $\widetilde{\boldsymbol{z}}'$, $\widetilde{\boldsymbol{z}}$ accordingly. Because $\mathrm{SoftMax}$ is 1-Lipschitz (Gao & Pavel, 2017, Proposition 4),

$$|q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \widetilde{\mathcal{D}}_n)| \leq \|W_o(\boldsymbol{z} - \widetilde{\boldsymbol{z}})\| \leq \|\boldsymbol{z} - \widetilde{\boldsymbol{z}}\|.$$

Using Lemma A.2 below, we further get

$$\big\|\boldsymbol{z} - \widetilde{\boldsymbol{z}}\big\| \leq \big\|\mathrm{LayerNorm}(\boldsymbol{u} + \boldsymbol{z}') - \mathrm{LayerNorm}(\widetilde{\boldsymbol{u}}' + \widetilde{\boldsymbol{z}})\big\| \leq 4\frac{|\gamma_1|}{|\gamma_2|}\Big(\big\|\boldsymbol{u} - \widetilde{\boldsymbol{u}}\big\| + \big\|\boldsymbol{z}' - \widetilde{\boldsymbol{z}}'\big\|\Big).$$

Because also ReLu is 1-Lipschitz,

$$\big\|\boldsymbol{z}' - \widetilde{\boldsymbol{z}}'\big\| = \big\|W_{r,2}\,\mathrm{ReLu}(W_{r,1}\boldsymbol{u}) - W_{r,2}\,\mathrm{ReLu}(W_{r,1}\widetilde{\boldsymbol{u}})\big\| \leq \|\boldsymbol{u} - \widetilde{\boldsymbol{u}}\|.$$

Using Lemma A.2 again,

$$\|\boldsymbol{u} - \widetilde{\boldsymbol{u}}\| = \|\,\mathrm{LayerNorm}(\boldsymbol{v} + \boldsymbol{u}') - \mathrm{LayerNorm}(\boldsymbol{v} + \widetilde{\boldsymbol{u}}')\| \leq 4\frac{|\gamma_1|}{|\gamma_2|}\|\boldsymbol{u}' - \widetilde{\boldsymbol{u}}'\|.$$

The last displays together yield

$$|q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \mathcal{D}_n) - q_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}, \widetilde{\mathcal{D}}_n)| \leq 32\frac{|\gamma_1|}{|\gamma_2|}\big\|\boldsymbol{u} - \widetilde{\boldsymbol{u}}\big\|. \tag{9}$$

Defining $\widetilde{\boldsymbol{V}}_i = \boldsymbol{V}_i$ for $i < n$, we obtain

$$\begin{aligned}
\frac{1}{H}\|\boldsymbol{u}' - \widetilde{\boldsymbol{u}}'\| &= \frac{1}{H}\Bigg\|\sum_{h=1}^{H}\Bigg[\sum_{j=1}^{n} a_j^{(h)} W_v^{(h)} \boldsymbol{V}_j - \sum_{j=1}^{n} \widetilde{a}_j^{(h)} W_v^{(h)} \widetilde{\boldsymbol{V}}_j\Bigg]\Bigg\| \\
&\leq \max_{1\leq h\leq H}\Bigg\|\sum_{j=1}^{n} a_j^{(h)} W_v^{(h)} \boldsymbol{V}_j - \sum_{j=1}^{n} \widetilde{a}_j^{(h)} W_v^{(h)} \widetilde{\boldsymbol{V}}_j\Bigg\| \\
&= \max_{1\leq h\leq H}\Bigg\|\sum_{j=1}^{n} a_j^{(h)} W_v^{(h)} (\boldsymbol{V}_j - \widetilde{\boldsymbol{V}}_j) + \sum_{j=1}^{n} (a_j^{(h)} - \widetilde{a}_j) W_v^{(h)} \widetilde{\boldsymbol{V}}_j\Bigg\| \\
&\leq \max_{1\leq h\leq H}\sum_{j=1}^{n} a_j^{(h)} \|W_v^{(h)}\| \|\boldsymbol{V}_j - \widetilde{\boldsymbol{V}}_j\| + \sum_{j=1}^{n} |\widetilde{a}_j^{(h)} - a_j| \|W_v^{(h)}\| \|\widetilde{\boldsymbol{V}}_j\| \\
&\leq 4 \max_{1\leq h\leq H} a_n^{(h)} + 2\sum_{j=1}^{n} |\widetilde{a}_j^{(h)} - a_j|.
\end{aligned}$$

Let $s_i = \boldsymbol{v}^\top W_q^{(h)} \boldsymbol{V}_i$ and note that $|s_i| \leq \|\boldsymbol{v}\| \|W_q^{(h)}\| \max_j \|\boldsymbol{V}_j\| \leq 4$.

$$|a_n^{(h)}| = \frac{\exp(s_n)}{\sum_{j=1}^{n}\exp(s_j)} \leq \frac{e^4}{\sum_{j=1}^{n} e^{-4}} = \frac{e^8}{n}.$$

Further,

$$\begin{aligned}
|\widetilde{a}_j^{(h)} - a_j^{(h)}| &= \left|\frac{\exp(s_j)}{\sum_{j=1}^{n}\exp(s_j)} - \frac{\exp(\widetilde{s}_j)}{\sum_{j=1}^{n}\exp(\widetilde{s}_j)}\right| \\
&= \left|\frac{\exp(s_j)}{\sum_{j=1}^{n}\exp(s_j)} - \frac{\exp(\widetilde{s}_j)}{\sum_{j=1}^{n}\exp(s_j)} + \frac{\exp(\widetilde{s}_j)}{\sum_{j=1}^{n}\exp(s_j)} - \frac{\exp(\widetilde{s}_j)}{\sum_{j=1}^{n}\exp(\widetilde{s}_j)}\right| \\
&\leq \left|\frac{\exp(s_j) - \exp(\widetilde{s}_j)}{\sum_{j=1}^{n}\exp(s_j)}\right| + \left|\frac{\exp(\widetilde{s}_j)}{\sum_{j=1}^{n}\exp(s_j)} - \frac{\exp(\widetilde{s}_j)}{\sum_{j=1}^{n}\exp(\widetilde{s}_j)}\right| \\
&= \left|\frac{\exp(s_j) - \exp(\widetilde{s}_j)}{\sum_{j=1}^{n}\exp(s_j)}\right| + \frac{\exp(\widetilde{s}_j)}{\sum_{j=1}^{n}\exp(s_j)}\left|\frac{\sum_{j=1}^{n}\exp(s_j) - \sum_{j=1}^{n}\exp(\widetilde{s}_j)}{\sum_{j=1}^{n}\exp(\widetilde{s}_j)}\right| \\
&= \left|\frac{\exp(s_j) - \exp(\widetilde{s}_j)}{\sum_{j=1}^{n}\exp(s_j)}\right| + \frac{\exp(\widetilde{s}_j)}{\sum_{j=1}^{n}\exp(s_j)}\left|\frac{\exp(s_n) - \exp(\widetilde{s}_n)}{\sum_{j=1}^{n}\exp(\widetilde{s}_j)}\right|.
\end{aligned}$$

The first term on the right is zero if $j \neq n$. For $j = n$, it can be bounded by $2e^8/n$ as before. Using the same argument for the second term above, we get

$$|\widetilde{a}_j^{(h)} - a_j^{(h)}| \leq \frac{2e^8}{n}\mathbb{1}(j = n) + \frac{2e^{16}}{n^2}.$$

Accordingly,

$$\frac{1}{H}\|\boldsymbol{u}' - \widetilde{\boldsymbol{u}}'\| \leq 4 \max_{1\leq h\leq H} a_n^{(h)} + 2\sum_{j=1}^{n} |\widetilde{a}_j^{(h)} - a_j| \leq \frac{4e^8}{n} + \frac{2e^8}{n} + n\frac{2e^{16}}{n^2} \leq \frac{8e^{16}}{n}. \tag{10}$$

Combining (9) and (10) proves the claim. □

Lemma A.2. *For any two vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^d$ it holds*

$$\big\|\mathrm{LayerNorm}(\boldsymbol{a};\boldsymbol{\gamma}) - \mathrm{LayerNorm}(\boldsymbol{b};\boldsymbol{\gamma})\big\| \leq 4\frac{|\gamma_1|}{|\gamma_2|}\|\boldsymbol{a} - \boldsymbol{b}\|.$$

Proof. Let us first assume $\mathrm{avg}(\boldsymbol{a}) = \mathrm{avg}(\boldsymbol{b}) = \mathbf{0}$ and, without loss of generality, $\|\boldsymbol{a}\| \geq \|\boldsymbol{b}\|$. It holds,

$$\begin{aligned}
&\big\|\mathrm{LayerNorm}(\boldsymbol{a};\boldsymbol{\gamma}) - \mathrm{LayerNorm}(\boldsymbol{b};\boldsymbol{\gamma})\big\| \\
&= |\gamma_1|\left\|\frac{\boldsymbol{a}}{\|\boldsymbol{a}\| + |\gamma_2|} - \frac{\boldsymbol{b}}{\|\boldsymbol{b}\| + |\gamma_2|}\right\| \\
&= |\gamma_1|\left\|\frac{\boldsymbol{a}}{\|\boldsymbol{a}\| + |\gamma_2|} - \frac{\boldsymbol{b}}{\|\boldsymbol{a}\| + |\gamma_2|} + \frac{\boldsymbol{b}}{\|\boldsymbol{a}\| + |\gamma_2|} - \frac{\boldsymbol{b}}{\|\boldsymbol{b}\| + |\gamma_2|}\right\| \\
&\leq |\gamma_1|\frac{\|\boldsymbol{a} - \boldsymbol{b}\|}{\|\boldsymbol{a}\| + |\gamma_2|} + |\gamma_1|\|\boldsymbol{b}\|\left|\frac{1}{\|\boldsymbol{a}\| + |\gamma_2|} - \frac{1}{\|\boldsymbol{b}\| + |\gamma_2|}\right| \\
&= |\gamma_1|\frac{\|\boldsymbol{a} - \boldsymbol{b}\|}{\|\boldsymbol{a}\| + |\gamma_2|} + |\gamma_1|\frac{\|\boldsymbol{b}\|}{\|\boldsymbol{b}\| + |\gamma_2|}\frac{\big|\|\boldsymbol{b}\| - \|\boldsymbol{a}\|\big|}{\|\boldsymbol{a}\| + |\gamma_2|} \\
&\leq |\gamma_1|\frac{\|\boldsymbol{a} - \boldsymbol{b}\|}{\|\boldsymbol{a}\| + |\gamma_2|} + |\gamma_1|\frac{\big|\|\boldsymbol{b}\| - \|\boldsymbol{a}\|\big|}{\|\boldsymbol{a}\| + |\gamma_2|} \\
&\leq 2\frac{|\gamma_1|\|\boldsymbol{a} - \boldsymbol{b}\|}{\|\boldsymbol{a}\| + |\gamma_2|} \qquad \text{[reverse triangle inequality]} \\
&\leq 2\frac{|\gamma_1|}{|\gamma_2|}\|\boldsymbol{a} - \boldsymbol{b}\|.
\end{aligned}$$

If $\mathrm{avg}(\boldsymbol{a}) \neq 0$ or $\mathrm{avg}(\boldsymbol{b}) \neq 0$, we get

$$\|\,\mathrm{LayerNorm}(\boldsymbol{a};\boldsymbol{\gamma}) - \mathrm{LayerNorm}(\boldsymbol{b};\boldsymbol{\gamma})\| \leq 2\frac{|\gamma_1|}{|\gamma_2|}\|\boldsymbol{a} - \boldsymbol{b} - \mathrm{avg}(\boldsymbol{a} - \boldsymbol{b})\| \leq 4\frac{|\gamma_1|}{|\gamma_2|}\|\boldsymbol{a} - \boldsymbol{b}\|,$$

because

$$\|\,\mathrm{avg}(\boldsymbol{a} - \boldsymbol{b})\|^2 = \left\|\frac{1}{d}\sum_{i=1}^{d}(a_i - b_i)\mathbf{1}\right\|^2 \leq \left(\frac{1}{d}\sum_{i=1}^{d}|a_i - b_i|\|\mathbf{1}\|\right)^2 = d\left(\frac{1}{d}\sum_{i=1}^{d}|a_i - b_i|\right)^2 \leq \|\boldsymbol{a} - \boldsymbol{b}\|^2,$$

where we used the triangle inequality in the second step and Jensen's inequality in the last. □

A.9. Proof of Theorem 6.3

We have

$$\boldsymbol{u}' = \sum_{h=1}^{H}\sum_{j=1}^{n} a_j^{(h)} W_v^{(h)}\boldsymbol{V}_j = \sum_{h=1}^{H}\frac{\frac{1}{n}\sum_{j=1}^{n}\exp(\boldsymbol{v}^\top W_q^{(h)}\boldsymbol{V}_j)W_v^{(h)}\boldsymbol{V}_j}{\frac{1}{n}\sum_{j=1}^{n}\exp(\boldsymbol{v}^\top W_q^{(h)}\boldsymbol{V}_j)}.$$

The law of large numbers implies $\boldsymbol{u}' \to \overline{\boldsymbol{u}}'$ almost surely, where

$$\overline{\boldsymbol{u}}' = \sum_{h=1}^{H} \frac{\mathbb{E}_{\boldsymbol{V}\sim p_0}[\exp(\boldsymbol{v}^\top W_q^{(h)} \boldsymbol{V}) W_v^{(h)} \boldsymbol{V}]}{\mathbb{E}_{\boldsymbol{V}\sim p_0}[\exp(\boldsymbol{v}^\top W_q^{(h)} \boldsymbol{V})]}.$$

Now, observe that

$$\frac{\mathbb{E}_{\boldsymbol{V}\sim p_0}[\exp(\boldsymbol{v}^\top W_q^{(h)} \boldsymbol{V}) W_v^{(h)} \boldsymbol{V}]}{\mathbb{E}_{\boldsymbol{V}\sim p_0}[\exp(\boldsymbol{v}^\top W_q^{(h)} \boldsymbol{V})]} = \int W_v^{(h)} \boldsymbol{s} \underbrace{\frac{\exp(\boldsymbol{v}^\top W_q^{(h)} \boldsymbol{s})}{\mathbb{E}_{\boldsymbol{V}\sim p_0}[\exp(\boldsymbol{v}^\top W_q^{(h)} \boldsymbol{V})]} p_0(\boldsymbol{s})}_{g_h(\boldsymbol{s})} \, d\boldsymbol{s} = W_v^{(h)} \mathbb{E}_{\boldsymbol{V}\sim g_h}[\boldsymbol{V}].$$

Since all following operations on $\boldsymbol{u}'$ are continuous (see the proof of Theorem A.1), it also holds

$$\lim_{n\to\infty} \left| q_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}, \mathcal{D}_n) - \overline{q}_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}) \right| = 0 \quad \text{almost surely.}$$

Because the sequence $|q_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}, \mathcal{D}_n) - q_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x})|$ is uniformly bounded by 2, convergence is also in L_1. This implies $\lim_{n\to\infty} \mathbb{E}[q_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}, \mathcal{D}_n)] = \overline{q}_\theta(\cdot \mid \boldsymbol{x})$. □

B. Approximation Results for the PFN Parameter

Suppose that the parameters $\boldsymbol{\theta}$ live in a subset of $\mathbb{R}^p$ with p fixed and finite. This assumption would be questionable in the usual machine learning setting, but our situation is different. A PFN is pre-trained offline, using m Monte-Carlo samples from data sets $\mathcal{D}^{(j)}$. Given enough computing power, we can take m as large as we want.

Given sufficient regularity, the following theorems are direct applications of existing results. To keep additional concepts and notation to a minimum, we omit detailed conditions and proofs and refer to the original works for specifics. We start with the behavior of $\widehat{\boldsymbol{\theta}}$.

Theorem B.1. *It holds:*

(i) $\widehat{\boldsymbol{\theta}} \to \boldsymbol{\theta}^*$ *in probability,*

(ii) $\sqrt{m}(\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*) \to \mathcal{N}(\mathbf{0}, \Sigma_{\boldsymbol{\theta}^*})$ *in distribution, where* $\Sigma_{\boldsymbol{\theta}^*} = I_{\boldsymbol{\theta}^*}^{-1} V_{\boldsymbol{\theta}^*} I_{\boldsymbol{\theta}^*}^{-1}$ *and*

$$\begin{aligned} I_{\boldsymbol{\theta}^*} &= \mathbb{E}_{\Pi_N} \mathbb{E}_{\Pi}[\nabla_{\boldsymbol{\theta}}^2 \log q_{\boldsymbol{\theta}^*}(Y \mid \boldsymbol{X}, \mathcal{D}_N)] \\ \textit{and} \quad V_{\boldsymbol{\theta}^*} &= \mathbb{E}_{\Pi_N} \mathbb{E}_{\Pi}[\nabla_{\boldsymbol{\theta}} \log q_{\boldsymbol{\theta}^*}(Y \mid \boldsymbol{X}, \mathcal{D}_N) \times \nabla_{\boldsymbol{\theta}}^\top \log q_{\boldsymbol{\theta}^*}(Y \mid \boldsymbol{X}, \mathcal{D}_N)]. \end{aligned}$$

Proof. See Corollary 3.2.3 and Example 3.2.12 in Vaart & Wellner (1996). □

The first part shows that $\widehat{\boldsymbol{\theta}}$ is a valid approximation of $\boldsymbol{\theta}^*$, the second quantifies its accuracy. Theorem B.1 has direct implications for the approximated model $q_{\widehat{\boldsymbol{\theta}}}$.

Theorem B.2. *It holds:*

(i) $\sup_{(y,\boldsymbol{x},\mathcal{D})} |q_{\widehat{\boldsymbol{\theta}}}(y \mid \boldsymbol{x}, \mathcal{D}) - q_{\boldsymbol{\theta}^*}(y \mid \boldsymbol{x}, \mathcal{D})| \to 0$ *in probability,*

(ii) $\sqrt{m}(q_{\widehat{\boldsymbol{\theta}}} - q_{\boldsymbol{\theta}^*})$ *converges weakly to a mean-zero Gaussian process with*

$$\mathbb{C}\mathrm{ov}\big((y \mid \boldsymbol{x}, \mathcal{D}), (y' \mid \boldsymbol{x}', \mathcal{D}')\big) = \nabla_{\boldsymbol{\theta}} q_{\boldsymbol{\theta}^*}(y \mid \boldsymbol{x}, \mathcal{D})^\top \Sigma_{\boldsymbol{\theta}^*} \nabla_{\boldsymbol{\theta}} q_{\boldsymbol{\theta}^*}(y' \mid \boldsymbol{x}', \mathcal{D}').$$

Proof. Part (i) follows from Theorem B.1 and the continuous mapping theorem (Vaart & Wellner, 1996, Theorem 1.11.1), part (ii) from the delta method (Vaart & Wellner, 1996, Theorem 3.9.4). □

From the second part, we see that the variance of $q_{\widehat{\boldsymbol{\theta}}}(y \mid \boldsymbol{x}, \mathcal{D})$ is approximately

$$\frac{1}{m} \nabla_{\boldsymbol{\theta}} q_{\boldsymbol{\theta}^*}(y \mid \boldsymbol{x}, \mathcal{D})^\top \Sigma_{\boldsymbol{\theta}^*} \nabla_{\boldsymbol{\theta}} q_{\boldsymbol{\theta}^*}(y \mid \boldsymbol{x}, \mathcal{D}).$$

Intuitively, the variance depends on the accuracy of $\widehat{\boldsymbol{\theta}}$ (through $\Sigma_{\boldsymbol{\theta}^*}$), and the sensitivity of $q_{\boldsymbol{\theta}^*}$ with respect to $\boldsymbol{\theta}^*$ (through $\nabla_{\boldsymbol{\theta}} q_{\boldsymbol{\theta}^*}$). Model complexity normally works against us in both parts. Hence, more complex models need to be trained with more Monte-Carlo samples to limit the variance.